IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:
Nisha D. Talagala
Chia Y. Wu

Serial No. 09/846,374

Filed: 4/30/2001

For: Data Integrity Error Handling in a Redundant Storage Array

§ Group Art Unit: 2133
§
§ Examiner: Trimmings, John P.
§
§ Atty. Dkt. No.: 5681-76700

CERTIFICATE OF MAILING
37 C.F.R. § 1.8
I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as First Class Mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below:

B. Noël Kivlin
Name of Registered Representative

September 29, 2004 _____
Date Signature

FEE AUTHORIZATION

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

The Commissioner is hereby authorized to charge the following fee to Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C. Deposit Account Number 50-1505/5681-76700:

Fee:	Appeal Brief
Amount:	$330.00
Attorney Docket No.:	5681-76700

The Commissioner is also authorized to charge any extension fee or other fees which may be necessary to the same account number.

Respectfully submitted,

B. Noël Kivlin
Reg. No. 33,929
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8800
Date: September 29, 2004

AF & FW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:
Nisha D. Talagala
Chia Y. Wu

Serial No. 09/846,374

Filed: 4/30/2001

For: Data Integrity Error Handling in a Redundant Storage Array

§ Group Art Unit: 2133
§
§ Examiner: Trimmings, John P.
§
§ Atty. Dkt. No.: 5681-76700
§ P5599

CERTIFICATE OF MAILING
37 C.F.R. § 1.8

I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as First Class Mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below:

B. Noël Kivlin
Name of Registered Representative

September 29, 2004
Date

Signature

APPEAL BRIEF

Box AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir/Madam:

Further to the Notice of Appeal filed July 29, 2004, Appellant presents this Appeal Brief. Appellant respectfully requests that this appeal be considered by the Board of Patent Appeals and Interferences.

10/06/2004 SSESHE1 00000003 501505 09846374
01 FC:1402 330.00 DA

I. REAL PARTY IN INTEREST

The subject application is owned by Sun Microsystems, Inc., a corporation organized and existing under and by virtue of the laws of the State of Delaware, and having its principal place of business at 4150 Network Circle, Santa Clara, CA 95054, as evidenced by the assignment recorded at Reel 0111784, Frame 0743.

II. RELATED APPEALS AND INTERFERENCES

No other appeals or interferences are known which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1-4, 6-19, 21-34, 36-49, and 51-60 are pending in the present application and are the subject of this appeal. Claims 1-4, 6-19, 21-34, 36-49, and 51-60 stand finally rejected under 35 U.S.C. § 112, first paragraph. A copy of Claims 1-4, 6-19, 21-34, 36-49, and 51-60, as on appeal (incorporating all amendments), is included in the Appendix hereto.

IV. STATUS OF AMENDMENTS

No amendment to the claims has been filed subsequent to the final rejection. The Appendix hereto reflects the current state of the claims.

V. SUMMARY OF CLAIMED SUBJECT MATTER

A storage array may include a plurality of storage devices utilizing a conventional RAID algorithm referred as RAID 5. RAID 5 may implement a striped disk array in which a segment of data is broken down into blocks and each block is written to a separate disk drive. An additional block, referred to as a parity block, may be calculated based on the values of the other blocks and may be written to a separate disk drive. The

parity block and the data blocks may be referred to as a parity group or a stripe, and the act of storing a segment of data across multiple disk drives may be referred to as striping. *See* page 7, paragraph [0021] and Figure 2. Each of the data blocks in this parity group may be described as having a redundancy relationship (i.e., horizontal redundancy relationship) with the other data blocks in that if one of the storage devices fails, the remaining storage devices may be used to reconstruct the data from the failed device. *See* pages 7-8, paragraph [0023] and Figure 2.

The term "vertical redundant relationship" may refer to a redundant relationship between a data block and a checksum corresponding to that data block. In contrast with the horizontal redundancy relationship, which exists between multiple data blocks, each vertical redundant relationship may exist between two segments of data: a checksum and a data block. *See* page 10, paragraph [0031] and Figure 3a.

In one particular embodiment, a process for handling data integrity errors may be implemented. First, a plurality of data blocks (e.g., five in this example) may be stored having a horizontal redundant relationship. Next, an array manager may generate the checksums, each checksum having a vertical redundant relationship with one of the stripe units. The checksums may be generated according to the various methodologies and may be stored in a variety of locations, including, for example, in a host, in a checksum memory provided in an array controller, in the array, or in a storage device separate from the array. In one example, the checksums may be appended to their corresponding stripe unit and stored with the stripe units on the disk drives. In another example, all of the checksums for array may be stored on a separate, dedicated checksum disk drive in storage system. *See* page 10-11, paragraphs [0033] and [0034] and Figures 4a and 4b.

Additionally, in one embodiment, a data integrity error may be detected in a first vertical redundant relationship. This data integrity error may be detected using various methods. In one embodiment, each time a host requests a data block from array, the checksum corresponding to that requested data block may also be retrieved and compared

to the data block. This comparison may be performed by calculating a new checksum for that data block and comparing that calculated new checksum to the original checksum. A data integrity error may be detected if there is a mismatch between the calculated checksum and the original checksum. In other embodiments, the checksums for all of the blocks of data on a disk drive may be reviewed periodically to search for data integrity errors. This periodic review may occur, for example, on a daily, weekly, or monthly basis in order to search for data integrity errors in data blocks which are infrequently accessed by host. *See* page 11, paragraph [0035] and Figures 4a and 4b.

The detection of a single data integrity error in one of the vertical redundant relationships can indicate that either the data block and/or its corresponding checksum contains an error. Then, a process may be implemented to provide a diagnosis and possible repair methodology for the error. *See* page 12, paragraph [0036] and Figures 4a-4e.

In one embodiment, after the data integrity error in one of the vertical redundant relationships is detected, all of the other vertical redundant relationships in that parity group may be reviewed. For example, if a data integrity error is detected in the vertical redundant relationship between a first checksum and a first stripe unit, the remaining checksums corresponding to the remaining stripe units of the stripe may then be checked against the data in the corresponding stripe units. *See* page 12, paragraph [0037] and Figures 4a-4e.

Then, the horizontal redundant relationship between the stripe units may be reviewed. If a total of three or more data integrity errors in the vertical redundant relationships are identified for that parity group, then a first exemplary process (i.e., the triple checksum error procedure of Figure 4c) for reviewing this horizontal redundant relationship may be implemented. If exactly two data integrity errors in the vertical redundant relationships are identified, then a second exemplary process (i.e., the double checksum error procedure of Figure 4d) may be implemented for reviewing this

horizontal redundant relationship. Finally, if only a single data integrity error is identified, then a third exemplary process (i.e., the single checksum error procedure of Figure 4e) for reviewing this horizontal redundant relationship may be implemented. *See* page 12, paragraph [0038] and Figures 4a-4e.

VI. GROUNDS OF REJECTION

1. Claims 1-4, 6-19, 21-34, 36-49, and 51-60 stand finally rejected under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement.

VII. ARGUMENT

A. Claims 1-4, 6-19, 21-34, 36-49, and 51-60

The Examiner rejected claims 1-4, 6-19, 21-34, 36-49, and 51-60 under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement. Appellant respectfully traverses these rejections in light of the following remarks.

The inquiry into whether the description requirement is met must be determined on a case-by-case basis and is a question of fact. *In re Wertheim*, 541 F.2d 257, 262, 191 USPQ 90, 96 (CCPA 1976). A description as filed is presumed to be adequate, unless or until sufficient evidence or reasoning to the contrary has been presented by the examiner to rebut the presumption. See, e.g., *In re Marzocchi*, 439 F.2d 220, 224, 169 USPQ 367, 370 (CCPA 1971). The examiner, therefore, must have a reasonable basis to challenge the adequacy of the written description. The examiner has the initial burden of presenting by a preponderance of evidence why a person skilled in the art would not recognize in an applicant's disclosure a description of the invention defined by the claims. *Wertheim*, 541 F.2d at 263, 191 USPQ at 97.

To satisfy the written description requirement, an applicant must convey with reasonable clarity to those skilled in the art that, as of the filing date sought, he or she was in possession of the invention, and that the invention, in that context, is whatever is now claimed. The test for sufficiency of support in a parent application is whether the disclosure of the application relied upon "reasonably conveys to the artisan that the inventor had possession at that time of the later claimed subject matter." *Ralston Purina Co. v. Far-Mar-Co., Inc.*, 772 F.2d 1570, 1575, 227 USPQ 177, 179 (Fed. Cir. 1985) (quoting *In re Kaslow*, 707 F.2d 1366, 1375, 217 USPQ 1089, 1096 (Fed. Cir. 1983)). An applicant shows possession of the claimed invention by describing the claimed invention with all of its limitations using such descriptive means as words, structures,

figures, diagrams, and formulas that fully set forth the claimed invention. *Lockwood v. American Airlines, Inc.,* 107 F.3d 1565, 1572, 41 USPQ2d 1961, 1966 (Fed. Cir. 1997).

Appellant respectfully disagrees with Examiner's assertion on page 4 of the Final Office Action dated April 29, 2004, stating, "Every embodiment of the invention responds to a 'detection' of errors, and not to 'indications' of errors (see specification Summary and Abstract)."

Contrary to the Examiner's assertion, Appellant discloses, on page 11, line 17 of Appellant's Specification, "In step 403, a data integrity error is detected in a first vertical redundant relationship." Appellant submits that when a data integrity error is detected in a vertical redundant relationship, it is unknown whether there is a data integrity error in the checksum or in the data block corresponding to the particular vertical redundant relationship. Appellant discloses, on page 12, lines 1-3 of Appellant's Specification, "The detection of a single data integrity error in one of the vertical redundant relationships **can indicate** that **either** the data block and/or its corresponding checksum contains an error." (Emphasis added) Therefore, the detection of a data integrity error in a vertical redundant relationship is an indication of a data integrity error in the corresponding data block; however, additional diagnosis may be necessary to determine whether the data integrity error is in the checksum and/or in the data block. Appellant discloses, on page 12, lines 3-4 of Appellant's Specification, "The process described herein with respect to FIGS. 4a-4e can provide a diagnosis and possible repair methodology for the error." (Emphasis added) For example, when a data integrity error is detected in exactly one of the vertical redundant relationships, the diagnosis to determine whether the checksum and/or the data block contains the error is disclosed on page 15, lines 13-18 of Appellant's Specification,

> "FIG. 4e illustrates the methodology used when a single checksum error condition is identified. In this situation, **either or both** the strip unit and its corresponding checksum are corrupted. In step 442, a stripe parity integrity test is performed to determine whether a data integrity error exists in the horizontal redundancy relationship 306. If there is no error in the

stripe parity, it is concluded that the stripe unit contains valid data and its checksum is restored in step 444." (Emphasis added)

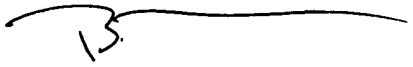
Accordingly, Appellant respectfully submits that the claimed features of Claims 1-4, 6-19, 21-34, 36-49, and 51-60 are supported by the specification. In view of the above arguments, Appellant respectfully requests reversal of the Examiner's 35 U.S.C. §112, first paragraph, rejections.

VIII. CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejections of claims 1-4, 6-19, 21-34, 36-49, and 51-60 were erroneous, and reversal of Examiner's decision is respectfully requested.

The Commissioner is authorized to charge the appeal brief fee of $330.00 and any other fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5681-76700/BNK.

Respectfully submitted,



B. Noël Kivlin
Reg. No. 33,929
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8800
Date: September 29, 2004

IX. CLAIMS APPENDIX

The claims on appeal are as follows.

1. A method of operating a storage system, comprising:

storing a plurality of data blocks having a horizontal redundant relationship;

storing a plurality of checksums, each checksum having a vertical redundant relationship with a corresponding one of the plurality of data blocks;

in response to an indication of a data integrity error in at least one of the plurality of data blocks, determining whether the vertical redundant relationship between each of the remaining data blocks and the corresponding checksums indicates an error; and

determining whether the horizontal redundant relationship between the plurality of data blocks indicates an error.

2. The method of Claim 1, wherein:

said determining whether the vertical redundant relationship between each of the remaining data blocks and the corresponding checksums indicates an error comprises calculating a new checksum for each of the remaining data blocks in the plurality of data blocks and comparing the calculated new checksum to the checksum from the plurality of checksums corresponding to that data block.

3. The method of Claim 1, wherein:

said storing the plurality of checksums comprises storing the plurality of checksums such that each data block in the plurality of data blocks has a vertical redundant relationship with a set of corresponding

checksums, and each of the checksums in the set of corresponding checksums has a horizontal checksum redundant relationship with the other checksums in the set of corresponding checksums.

4. The method of Claim 3, further comprising:

in response to detecting a first data integrity error between a first data block and a first checksum from a first set of checksums corresponding to the first data block, reviewing the vertical redundant relationship between the first data block and the remaining checksums in the first set of checksums; and

in response to detecting no data integrity errors between the first data block and the remaining checksums in the first set of checksums, creating a restored first checksum to replace the first checksum corresponding to the first data integrity error.

6. The method of Claim 1, wherein:

said plurality of data blocks comprises a plurality of data stripe units and a parity stripe unit; and

said determining whether the horizontal redundant relationship between the plurality of data blocks indicates an error comprises calculating a new parity stripe unit based on the plurality of data stripe units and comparing the calculated new parity stripe unit to the parity stripe unit from the plurality of data blocks.

7. The method of Claim 1, wherein:

said plurality of data blocks comprises a first mirrored data block and a corresponding second mirrored data block; and

said determining whether the horizontal redundant relationship between the plurality of data blocks indicates an error

comprises comparing the first mirrored data block and the second mirrored data block.

8. The method of Claim 1, further comprising:

in response to detecting a first data integrity error in exactly one of the vertical redundant relationships and detecting no data integrity errors in the horizontal redundant relationship, creating a restored checksum using the data block corresponding to the first data integrity error to replace the existing checksum corresponding to the first data integrity error.

9. The method of Claim 1, further comprising:

in response to detecting a first data integrity error in exactly one of the vertical redundant relationships and detecting a second data integrity error in the horizontal redundant relationship, creating a restored data block using the horizontal redundant relationship to replace the data block corresponding to the first data integrity error and comparing the restored data block to the checksum corresponding to the first data integrity error.

10. The method of Claim 9, further comprising:

in response to a mismatch between the restored data block and the checksum corresponding to the first data integrity error, creating a restored checksum using the restored data block to replace the checksum corresponding to the first data integrity error.

11. The method of Claim 1, further comprising:

in response to detecting a first data integrity error in a first vertical redundant relationship, detecting a second data integrity error in a second vertical redundant relationship, and detecting no data

integrity errors in the horizontal redundant relationship, reporting a data loss condition.

12. The method of Claim 1, further comprising:

in response to detecting a first data integrity error in a first vertical redundant relationship, detecting a second data integrity error in a second vertical redundant relationship, and detecting a third data integrity error in the horizontal redundant relationship:

creating a first restored data block using the horizontal redundant relationship to replace the data block corresponding to the first vertical redundant relationship;

comparing the first restored data block to the checksum corresponding to the first vertical redundant relationship;

creating a second restored data block using the horizontal redundant relationship to replace the data block corresponding to the second vertical redundant relationship; and

comparing the second restored data block to the checksum corresponding to the second vertical redundant relationship.

13. The method of Claim 12, further comprising:

reporting a data loss condition in response to either:

(a) a mismatch between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a mismatch between the second restored data block and the checksum corresponding to the second vertical redundant relationship; or

(b) a match between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a

match between the second restored data block and the checksum corresponding to the second vertical redundant relationship.

14. The method of Claim 12, further comprising:

in response to a match between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a mismatch between the second restored data block and the checksum corresponding to the second vertical redundant relationship, creating a restored checksum using the second restored data block to replace the checksum corresponding to the second vertical redundant relationship.

15. The method of Claim 1, further comprising:

in response to detecting data integrity errors in at least three of the vertical redundant relationships, reporting a data loss.

16. A storage system, comprising:

a storage array comprising a plurality of storage devices;

a plurality of data blocks having a horizontal redundant relationship, each data block stored on one of the plurality of storage devices;

a plurality of checksums, each checksum having a vertical redundant relationship with a corresponding one of the plurality of data blocks; and

an array manager coupled to the storage array, wherein said array manager is configured such that in response to an indication of a data integrity error in at least one of the plurality of data blocks, said array manager determines whether the vertical redundant relationship between each of the remaining data blocks and the corresponding checksums indicates an error;

wherein the array manager further determines whether the horizontal redundant relationship between the plurality of data blocks indicates an error.

17. The storage system of Claim 16, wherein:

said array manager is configured to determine whether the vertical redundant relationship between each of the remaining data blocks and the corresponding checksums indicates an error by calculating a new checksum for each of the remaining data blocks in the plurality of data blocks and comparing the calculated new checksum to the checksum from the plurality of checksums corresponding to that data block.

18. The storage system of Claim 16, wherein:

each data block in the plurality of data blocks has a vertical redundant relationship with a set of corresponding checksums, and each of the checksums in the set of corresponding checksums has a horizontal checksum redundant relationship with the other checksums in the set of corresponding checksums.

19. The storage system of Claim 18, wherein:

said array manager is configured such that in response to detecting a first data integrity error between a first data block and a first checksum from a first set of checksums corresponding to the first data block, said array manager reviews the vertical redundant relationship between the first data block and the remaining checksums in the first set of checksums; and

said array manager is further configured such that in response to detecting no data integrity errors between the first data block and the remaining checksums in the first set of checksums, said array manager creates a restored first checksum to replace the first checksum corresponding to the first data integrity error.

21. The storage system of Claim 16, wherein:

said plurality of data blocks comprises a plurality of data stripe units and a parity stripe unit; and

said array manager is configured to determine whether the horizontal redundant relationship between the plurality of data blocks indicates an error by calculating a new parity stripe unit based on the plurality of data stripe units and comparing the calculated new parity stripe unit to the parity stripe unit from the plurality of data blocks.

22. The storage system of Claim 16, wherein:

said plurality of data blocks comprises a first mirrored data block and a corresponding second mirrored data block; and

said array manager is configured to determine whether the horizontal redundant relationship between the plurality of data blocks indicates an error by comparing the first mirrored data block and the second mirrored data block.

23. The storage system of Claim 16, wherein:

said array manager is configured such that in response to detecting a first data integrity error in exactly one of the vertical redundant relationships and detecting no data integrity errors in the horizontal redundant relationship, said array manager creates a restored checksum using the data block corresponding to the first data integrity error to replace the existing checksum corresponding to the first data integrity error.

24. The storage system of Claim 16, wherein:

said array manager is configured such that in response to detecting a first data integrity error in exactly one of the vertical redundant relationships and detecting a second data integrity error in the horizontal redundant relationship, said array manager creates a restored data block using the horizontal redundant relationship to replace the data block corresponding to the first data integrity error and compares the restored data block to the checksum corresponding to the first data integrity error.

25. The storage system of Claim 24, wherein:

said array manager is configured such that in response to a mismatch between the restored data block and the checksum corresponding to the first data integrity error, said array manager creates a restored checksum using the restored data block to replace the checksum corresponding to the first data integrity error.

26. The storage system of Claim 16, further comprising:

said array manager is configured such that in response to detecting a first data integrity error in a first vertical redundant relationship, detecting a second data integrity error in a second vertical redundant relationship, and detecting no data integrity errors in the horizontal redundant relationship, said array manager reports a data loss condition.

27. The storage system of Claim 16, wherein:

said array manager is configured such that in response to detecting a first data integrity error in a first vertical redundant relationship, detecting a second data integrity error in a second vertical redundant relationship, and detecting a third data integrity error in the horizontal redundant relationship, said array manager:

creates a first restored data block using the horizontal redundant relationship to replace the data block corresponding to the first vertical redundant relationship;

compares the first restored data block to the checksum corresponding to the first vertical redundant relationship;

creates a second restored data block using the horizontal redundant relationship to replace the data block corresponding to the second vertical redundant relationship; and

compares the second restored data block to the checksum corresponding to the second vertical redundant relationship.

28. The storage system of Claim 27, wherein:

said array manager is configured such that said array manager reports a data loss condition in response to either:

(a) a mismatch between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a mismatch between the second restored data block and the checksum corresponding to the second vertical redundant relationship; or

(b) a match between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a match between the second restored data block and the checksum corresponding to the second vertical redundant relationship.

29. The storage system of Claim 27, wherein:

said array manager is configured such that in response to a match between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a mismatch between the second restored data block and the checksum corresponding to the second vertical redundant relationship, said array manager creates a restored checksum using the second restored data block to replace the checksum corresponding to the second vertical redundant relationship.

30. The storage system of Claim 16, wherein:

said array manager is configured to report a data loss condition in response to detecting data integrity errors in at least three of the vertical redundant relationships.

31. A computer-readable medium whose contents cause a computer system to operate a storage system by performing the steps of:

detecting an indication of a data integrity error in at least one of a plurality of data block having a horizontal redundant relationship;

analyzing a plurality of checksums, each checksum having a vertical redundant relationship with a corresponding one of the plurality of data blocks, said analyzing comprising determining whether the vertical redundant relationship between each of the remaining data blocks and the corresponding checksums indicates an error; and

determining whether the horizontal redundant relationship between the plurality of data blocks indicates an error.

32. The computer-readable medium of Claim 31, wherein:

said determining whether the vertical redundant relationship between each of the remaining data blocks and the corresponding checksums indicates an error comprises calculating a new checksum for each of the remaining data blocks in the plurality of data blocks and comparing the calculated new checksum to the checksum from the plurality of checksums corresponding to that data block.

33. The computer-readable medium of Claim 31, wherein:

each data block in the plurality of data blocks has a vertical redundant relationship with a set of corresponding checksums, and each of the checksums in the set of corresponding checksums has a horizontal checksum redundant relationship with the other checksums in the set of corresponding checksums.

34. The computer-readable medium of Claim 33, wherein the contents of the computer--readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to detecting a first data integrity error between a first data block and a first checksum from a first set of checksums corresponding to the first data block, reviewing the vertical redundant relationship between the first data block and the remaining checksums in the first set of checksums; and

in response to detecting no data integrity errors between the first data block and the remaining checksums in the first set of checksums, creating a restored first checksum to replace the first checksum corresponding to the first data integrity error.

36. The computer-readable medium of Claim 31, wherein:

said plurality of data blocks comprises a plurality of data stripe units and a parity stripe unit; and

said determining whether the horizontal redundant relationship between the plurality of data blocks indicates an error comprises calculating a new parity stripe unit based on the plurality of data stripe units and comparing the calculated new parity stripe unit to the parity stripe unit from the plurality of data blocks.

37. The computer-readable medium of Claim 31, wherein:

said plurality of data blocks comprises a first mirrored data block and a corresponding second mirrored data block; and

said determining whether the horizontal redundant relationship between the plurality of data blocks indicates an error comprises comparing the first mirrored data block and the second mirrored data block.

38. The computer-readable medium of Claim 31, wherein the contents of the computer-readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to detecting a first data integrity error in exactly one of the vertical redundant relationships and detecting no data integrity errors in the horizontal redundant relationship, creating a restored checksum using the data block corresponding to the first data integrity error to replace the existing checksum corresponding to the first data integrity error.

39. The computer-readable medium of Claim 31, wherein the contents of the computer--readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to detecting a first data integrity error in exactly one of the vertical redundant relationships and detecting a second data integrity error in the horizontal redundant relationship, creating a restored data block using the horizontal redundant relationship to replace the data block corresponding to the first data integrity error and comparing the restored data block to the checksum corresponding to the first data integrity error.

40. The computer-readable medium of Claim 39, wherein the contents of the computer-readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to a mismatch between the restored data block and the checksum corresponding to the first data integrity error, creating a restored checksum using the restored data block to replace the checksum corresponding to the first data integrity error.

41. The computer-readable medium of Claim 31, wherein the contents of the computer--readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to detecting a first data integrity error in a first vertical redundant relationship, detecting a second data integrity error in a second vertical redundant relationship, and detecting no data integrity errors in the horizontal redundant relationship, reporting a data loss condition.

42. The computer-readable medium of Claim 31, wherein the contents of the computer-readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to detecting a first data integrity error in a first vertical redundant relationship, detecting a second data integrity error in a second vertical redundant relationship, and detecting a third data integrity error in the horizontal redundant relationship:

creating a first restored data block using the horizontal redundant relationship to replace the data block corresponding to the first vertical redundant relationship;

comparing the first restored data block to the checksum corresponding to the first vertical redundant relationship;

creating a second restored data block using the horizontal redundant relationship to replace the data block corresponding to the second vertical redundant relationship; and

comparing the second restored data block to the checksum corresponding to the second vertical redundant relationship.

43. The computer-readable medium of Claim 42, wherein the contents of the computer--readable medium cause the computer system to operate the storage system by performing the further steps of:

reporting a data loss condition in response to either:

(a) a mismatch between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a mismatch between the second restored data block and the

checksum corresponding to the second vertical redundant relationship; or

(b) a match between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a match between the second restored data block and the checksum corresponding to the second vertical redundant relationship.

44. The computer-readable medium of Claim 42, wherein the contents of the computer-readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to a match between the first restored data block and the checksum corresponding to the first vertical redundant relationship and a mismatch between the second restored data block and the checksum corresponding to the second vertical redundant relationship, creating a restored checksum using the second restored data block to replace the checksum corresponding to the second vertical redundant relationship.

45. The computer-readable medium of Claim 31, wherein the contents of the computer-readable medium cause the computer system to operate the storage system by performing the further steps of:

in response to detecting data integrity errors in at least three of the vertical redundant relationships, reporting a data loss.

46. A storage system, comprising:

a plurality of storage devices configured to store a plurality of data blocks and a plurality of checksums, wherein each data block in said plurality of data blocks has a first type of redundant relationship with a corresponding one of the plurality of checksums and each data block in said plurality of data blocks has a second type of redundant relationship with the remaining data blocks in said plurality of data blocks; and

an array manager coupled to said plurality of storage devices, wherein said array manager is configured such that in response to an indication of a data integrity error in at least one of the plurality of data blocks, said array manager determines whether the first type of redundant relationships between each of the remaining data blocks and the corresponding checksums indicates an error;

wherein the array manager further determines whether the second type of redundant relationship between the plurality of data blocks indicates an error.

47. The storage system of Claim 46, wherein:

said array manager is configured to determine whether the first type of redundant relationships between each of the remaining data blocks and the corresponding checksums indicates an error by calculating a new checksum for each of the remaining data blocks in the plurality of data blocks and comparing the calculated new checksum to the checksum from the plurality of checksums corresponding to that data block.

48. The storage system of Claim 46, wherein:

each data block in the plurality of data blocks has a vertical redundant relationship with a set of corresponding checksums, and each of the checksums in the set of corresponding checksums has a horizontal checksum redundant relationship with the other checksums in the set of corresponding checksums.

49. The storage system of Claim 48, wherein:

said array manager is configured such that in response to detecting a first data integrity error between a first data block and a first checksum from a first set of checksums corresponding to the first data block, said array manager

reviews the redundant relationships between the first data block and the remaining checksums in the first set of checksums; and

said array manager is further configured such that in response to detecting no data integrity errors between the first data block and the remaining checksums in the first set of checksums, said array manager creates a restored first checksum to replace the first checksum corresponding to the first data integrity error.

51. The storage system of Claim 46, wherein:

said plurality of data blocks comprises a plurality of data stripe units and a parity stripe unit; and

said array manager is configured to determine whether the second type of redundant relationship between the plurality of data blocks indicates an error by calculating a new parity stripe unit based on the plurality of data stripe units and comparing the calculated new parity stripe unit to the parity stripe unit from the plurality of data blocks.

52. The storage system of Claim 46, wherein:

said plurality of data blocks comprises a first mirrored data block and a corresponding second mirrored data block; and

said array manager is configured to determine whether the second type of redundant relationship between the plurality of data blocks indicates an error by comparing the first mirrored data block and the second mirrored data block.

53. The storage system of Claim 46, wherein:

said array manager is configured such that in response to detecting a first data integrity error in exactly one of the first type of redundant relationships and detecting no data integrity errors in the second type of redundant relationship, said array manager creates a restored checksum using the

data block corresponding to the first data integrity error to replace the existing checksum corresponding to the first data integrity error.

54. The storage system of Claim 46, wherein:

said array manager is configured such that in response to detecting a first data integrity error in exactly one of the first type of redundant relationships and detecting a second data integrity error in the second type of redundant relationship, said array manager creates a restored data block using the second type of redundant relationship to replace the data block corresponding to the first data integrity error and compares the restored data block to the checksum corresponding to the first data integrity error.

55. The storage system of Claim 54, wherein:

said array manager is configured such that in response to a mismatch between the restored data block and the checksum corresponding to the first data integrity error, said array manager creates a restored checksum using the restored data block to replace the checksum corresponding to the first data integrity error.

56. The storage system of Claim 46, further comprising:

said array manager is configured such that in response to detecting a first data integrity error in a first one of the first type of redundant relationship, detecting a second data integrity error in a second one of the first type of redundant relationship, and detecting no data integrity errors in the second type of redundant relationship, said array manager reports a data loss condition.

57. The storage system of Claim 46, wherein:

said array manager is configured such that in response to detecting a first data integrity error in a first one of the first type of redundant relationship,

detecting a second data integrity error in a second one of the first type of redundant relationship, and detecting a third data integrity error in the second type of redundant relationship, said array manager:

creates a first restored data block using the second type of redundant relationship to replace the data block corresponding to the first one of the first type of redundant relationship;

compares the first restored data block to the checksum corresponding to the first one of the first type of redundant relationship;

creates a second restored data block using the second type of redundant relationship to replace the data block corresponding to the second one of the first type of redundant relationship; and

compares the second restored data block to the checksum corresponding to the second one of the first type of redundant relationship.

58. The storage system of Claim 57, wherein:

said array manager is configured such that said array manager reports a data loss condition in response to either:

(a) a mismatch between the first restored data block and the checksum corresponding to the first one of the first type of redundant relationship and a mismatch between the second restored data block and the checksum corresponding to the second one of the first type of redundant relationship;

(b) a match between the first restored data block and the checksum corresponding to the first one of the first type of redundant relationship and a match between the second restored data block and the checksum corresponding to the second one of the first type of redundant relationship.

59. The storage system of Claim 57, wherein:

said array manager is configured such that in response to a match between the first restored data block and the checksum corresponding to the first one of the first type of redundant relationship and a mismatch between the second restored data block and the checksum corresponding to the second one of the first type of redundant relationship, said array manager creates a restored checksum using the second restored data block to replace the checksum corresponding to the second one of the first type of redundant relationship.

60. The storage system of Claim 46, wherein:

said array manager is configured to report a data loss condition in response to detecting data integrity errors in at least three of the first type of redundant relationships.

X. EVIDENCE APPENDIX

None

X. RELATED PROCEEDINGS APPENDIX

None